Getting and Cleaning Data - Week 2

KR

2/9/2022

Reading from MySQL

```
knitr::opts_chunk$set(echo = T,
                      results = "hide",
                      warning = FALSE,
                      message = TRUE)
```

- A default chunk option to show code in the markdown file without printing the output
  {r setup, include=FALSE} knitr::opts_chunk$set(echo = T, results = "hide")

Individual chunks can be modified if necessary

{r analysis, results="markup"}

- Sometimes warnings appear that may not affect the code or can be safely ignored. To remove these warning messages from the markdown file, use:
  knitr::opts_chunk$set(warning = FALSE, message = FALSE)

```
# Load package

library(RMySQL)
```

```
## Loading required package: DBI
```

```
# Connect to database, send commands to database, disconnect from database

ucscDb <- dbConnect(MySQL(), user="genome", db="hg19",
                    host="genome-mysql.cse.ucsc.edu")
result <- dbGetQuery(ucscDb, "show databases;"); dbDisconnect(ucscDb);
```

- **dbConnect** - (command) connect to databases, in this case it is a MySQL database
- Host - where the database (MySQL) can be found
- ucscDb - handle assigned to the connection - always assign each connection a handle
- **dbGetQuery** - (command) used to pass commands to the database.
  In the example, "show databases" is a MySQL command that is sent to the database through the dbGetQuery function.
- **dbDisconnect** - (command) disconnect from the server (MySQL)
  Always important to disconnect from the server when finished retieving or analysing data
- " [1] True " confirms that we disconnected from the server

```
result

# Returns a list of all the available databases located at the host address indicated above
```

```
# Accessing a particular database, in this case hg19, within the MySQL server "host=..."

hg19 <- dbConnect(MySQL(), user="genome", db="hg19",
                  host="genome-mysql.cse.ucsc.edu")


# To determine what tables are in the database and how many tables there are

allTables <- dbListTables(hg19)
length(allTables)

allTables[1:5]
```

Getting Dimensions of a Specific Table

```
dbListFields(hg19, "affyU133Plus2")

# Lists fields in the table "affyU133Plus2", found in the hg19 database
```

```
dbGetQuery(hg19, "select count(*) from affyU133Plus2")

# Returns the number of elements in that table
```

Reading from the Table

```
affyData <- dbReadTable(hg19, "affyU133Plus2")

head(affyData)

# Reads the table or data frame for affyU133Plus2
```

Selecting a Specific Subset

```
query <- dbSendQuery(hg19, "select * from affyU133Plus2 where misMatches between 1 and 3")

affyMis <- fetch(query); quantile(affyMis$misMatches)
# Selects a subset of the data

affyMisSmall <- fetch(query, n=10); dbClearResult(query);
```

- If you only want a small subset of the data (instead of, say, an entire table), you can specify : e.g. fetch(query, n=10) to return only the first 10 rows
- After sending a query (dbSendQuery), you need to clear the query (dbClearResult(query))

```
dim(affyMisSmall)
```

Close the Connection

```
dbDisconnect(hg19)
```

- Remember to close the server as soon as you do not need it anymore, i.e. when you're finished retrieving or analysing the data

- Might be useful: http://www.r-bloggers.com/mysql-and-r/